# BRVO: Predicting Pedestrian Trajectories using Velocity-Space Reasoning

Sujeong Kim Stephen J. Guy Wenxi Liu David Wilkie Rynson W. H. Lau Ming C. Lin
Dinesh Manocha
http://gamma.cs.unc.edu/BRVO/

## Abstract

We introduce a novel, online method to predict pedestrian trajectories using agent-based velocity-space reasoning for improved human-robot interaction and collision-free navigation. Our formulation uses velocity obstacles to model the trajectory of each moving pedestrian in a robot's environment and improves the motion model by adaptively learning relevant parameters based on sensor data. The resulting motion model for each agent is computed using statistical inferencing techniques, including a combination of Ensemble Kalman filters and a maximum-likelihood estimation algorithm. This allows a robot to learn individual motion parameters for every agent in the scene at interactive rates. We highlight the performance of our motion prediction method in real-world crowded scenarios, compare its performance with prior techniques, and demonstrate the improved accuracy of the predicted trajectories. We also adapt our approach for collision-free robot navigation among pedestrians based on noisy data and highlight the results in our simulator.

## 1 Introduction

Robots are becoming increasingly common in everyday life. As more robots are introduced into human surroundings, it becomes increasingly important to develop safe and reliable techniques for human-robot interaction. Robots working around humans must be able to successfully navigate to their goal positions in dynamic environments with multiple people moving around them. A robot in a dynamic environment thus needs the ability to sense, track, and predict the position of all people moving in its workspace to navigate complicated environments without collisions.

Sensing and tracking the position of moving humans has been studied in robotics and computer vision, e.g. [Luber et al. 2010; Rodriguez et al. 2009; Kratz and Nishino 2011]. These methods often depend upon an a priori motion model fitted for the scenario in question. However, such motion priors are typically based on globally-optimized parameters for all the trajectories during the entire sequence of data, rather than only taking into account the specific individual's movements. As a result, current motion models are not able to accurately capture or predict the trajectory of each pedestrian. For example, we frequently observe atypical pedestrian motions, such as moving against the flow of other agents in a crowd, or quick velocity changes to avoid collisions. In order to address these issues, many pedestrian tracking algorithms use multi-agent or crowd motion models based on local collision avoidance [Pellegrini et al. 2009; Yamaguchi et al. 2011]. These multi-agent interaction models effectively capture short-term deviations from goal-directed paths, but in order to do so, they must already know each pedestrian's destination; they often use handpicked destination information, or other heuristics that require prior knowledge about the environment. As a result, these techniques have many limitations: they are unable to account for unknown environments with multiple destinations, or times when pedestrians take long detours or make unexpected stops. In general, the assumption that the destination information remains constant can often result in large errors in predicted trajectories. Moreover, environmental factors, such as static or dynamic obstacles, dense crowds, and social/cultural behavior rules, make predicting the behavior or movement of each individual pedestrian more complicated, thereby making it more difficult to compute a motion from a single destination.

In this work, we seek to overcome these limitations by presenting a novel online prediction method (BRVO) that is built on agent-based, velocity-space reasoning combined with Bayesian statistical inference; BRVO can provide an individualized motion model for each agent in a robot's environment. Our approach models each pedestrian's motion with their own unique characteristics, while also taking into account interactions with other pedestrians. This can be contrasted with the previous methods that use general motion priors or simple motion models. Moreover, our formulation is capable of dynamically adjusting the motion model for each individual in the presence of sensor noise and model uncertainty. We address the problems associated with fixed destination by integrating learning into our predictive framework and by adjusting short-term steering information.

Our approach works as follows. We generalize the problem by assuming that at any given time the robot has past observations for each person in the environment and wants to predict agent motion in the next several timesteps (i.e., to aid in navigation or planning tasks). We apply Ensemble Kalman Filtering (EnKF) to estimate the parameters for a human motion model based on *Reciprocal Velocity Obstacles* (RVO) [van den Berg et al. 2008a; van den Berg et al. 2011]. We use this combination of filtering and parameter estimation in crowded environments to infer the most likely state for each observed person: its position, velocity, and goal velocity. Based on the estimated parameters, we can predict the trajectory of each person in the environment, as well as their goal position. Our experiments with real-world pedestrian datasets demonstrate that our online per-agent learning method generates more accurate motion predictions than prior methods, especially for complex scenarios with noisy, low-resolution sensors, and missing data. Using our model, we also present an algorithm to compute collision-free trajectories for robots, which takes into account kinematic constraints; our approach can compute these paths at interactive rates in scenarios with dozens of pedestrians.

The rest of our paper is organized as follows. Section 2 reviews related work. Section 3 provides an overview of motion-prediction methods and the RVO crowd simulation method. Section 4 describes how our approach combines an adaptive machine-learning framework with RVO, and Section 5 presents our emperical results using real-world (video recorded) data, along with analysis on these results. Section 6 presents a method to integrate our pedestrian pre-

---

`http://gamma.cs.unc.edu/BRVO/`

Sujeong Kim, David Wilkie, Ming C. Lin, and Dinesh Manocha are with the Department of Computer Science, University of North Carolina at Chapel Hill. E-mail: {sujeong, wilkie, lin, dm}@cs.unc.edu

Stephen J. Guy is with the Department of Computer Science, University of Minnesota. E-mail: sjguy@cs.umn.edu

Wenxi Liu and Rynson W. H. Lau are with the Department of Computer Science, City University of Hong Kong. E-mail: wenxiliu2@student.cityu.edu.hk, rynson.lau@cityu.edu.hk

A preliminary version of this paper appeared in [Kim et al. 2012]

diction approach with modern mobile robot navigation techniques to improve collision avoidance for robots navigating in environments with moving pedestrians.

# 2 Related Work

In this section, we give an overview of prior work on motion models in robotics and crowd simulation and on their applications to pedestrian tracking and robot navigation.

## 2.1 Motion Models

There is an extensive body of work in robotics, multi-agent systems, crowd simulation, and computer vision on modeling pedestrians' motion in crowded environments. Here we provide a brief review of some recent work in this field. Many motion models have come from the fields of pedestrian dynamics and crowd simulation [Schadschneider et al. 2009; Pelechano et al. 2008]. These models are broadly classifiable into few main categories: potential-based models, which models virtual agents in a crowd as particles with potentials and forces [Helbing and Molnar 1995]; boid-like approaches, which create simple rules to steer agents [Reynolds 1999]; geometric models, which compute collision-free velocities [van den Berg et al. 2008b; van den Berg et al. 2011]; and field-based methods, which generate fields based on continuum theory or fluid models [Treuille et al. 2006]. Among these approaches, velocity-based motion models [Karamouzas et al. 2009; Karamouzas and Overmars 2010; van den Berg et al. 2008b; van den Berg et al. 2011; Pettré et al. 2009] have been successfully applied to the simulation and analysis of crowd behaviors and to multi-robot coordination [Snape et al. 2011]; velocity-based models have also been shown to have efficient implementations that closely match real human paths [Guy et al. 2010].

## 2.2 People-Tracking with Motion Models

Much of the previous work in people-tracking, including [Fod et al. 2002; Schulz et al. 2003; Cui et al. 2005], attempts to improve tracking quality by making simple assumptions about pedestrian motion. In recent years, robotics and computer vision literature have developed more sophisticated pedestrian motion models. For example, long-term motion planning models have been proposed to combine with tracking. Bruce and Gordon [Bruce and Gordon 2004] and Gong et al. [Gong et al. 2011] propose methods to first estimate pedestrians' destinations and then predict their motions along the path towards the estimated goal positions. Liao et al. propose a method to extract a Voronoi graph from the environment and predict people's motion along the edges, following the topological shape of the environment [Liao et al. 2003]. Methods that use short-term motion models for people-tracking are also an active area of development. Luber et al. apply Helbing's social force model to track people using a Kalman-filter based tracker [Luber et al. 2010] . Mehran et al. also apply the social force model to detect people's abnormal behaviors from video [Mehran et al. 2009]. Pellegrini et al. use an energy function to build up a goal-directed short-term collision-avoidance motion model, that they call Linear Trajectory Avoidance (LTA), to improve the accuracy of their people-tracking from video data [Pellegrini et al. 2009]. More recently, Yamaguchi et al. present a pedestrian-tracking algorithm that uses an agent-based behavioral model called ATTR, with additional social and personal properties learned from the behavioral priors, such as grouping information and destination information [Yamaguchi et al. 2011].

## 2.3 Robot Navigation in Crowds

Robots navigating in complex, noisy, and dynamic environments have prompted the development of other forms of trajectory prediction. For example, [Fulgenzi et al. 2007] use a probabilistic velocity-obstacle approach combined with the dynamic occupancy grid; this method's robot navigation, however, assumes the constant linear velocity motion of the obstacles, which is not always borne out in real-world data. [Du Toit and Burdick 2010] present a robot planning framework that takes into account pedestrians' anticipated future location information to reduce the uncertainty of the predicted belief states. Other work uses potential-based approaches for robot path planning in dynamic environments [Pradhan et al. 2011; Svenstrup et al. 2010].

Some methods use collected data to learn the trajectories. Ziebart et al. use pedestrian trajectories collected in the environment for prediction [Ziebart et al. 2009]. [Henry et al. 2010] use reinforced learning from example traces, estimating pedestrian density and flow with a Gaussian process. Bennewitz et al. apply Expectation Maximization clustering to learn typical motion patterns from pedestrian trajectories, and then guide a mobile robot using Hidden Markov Models to predict future pedestrian motion [Bennewitz et al. 2005]. Broadly speaking, our work differs from these approaches in that we combine the established pedestrian simulation method (RVO) with online learning to produce individualized motion predictions for each agent.

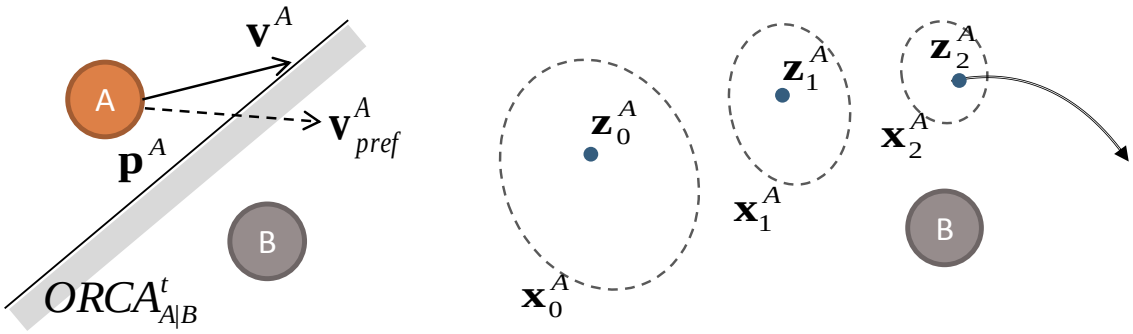
# 3 Background

In this section, we give an overview of motion-prediction methods, including offline and online methods. We use the term 'offline prediction methods' to refer to the techniques that perform some preprocessing, either manual or automatic, of the same scenarios [Gong et al. 2011; Pellegrini et al. 2009; Yamaguchi et al. 2011; Rodriguez et al. 2009; Kratz and Nishino 2011]; in other words, offline methods use global knowledge about past, present, and future inputs. We use the term 'online prediction methods' to refer to the techniques that use past observed data only without using information learned from the scene, prior to the prediction. For example, online people/object tracking systems can be designed using filtering-based methods (e.g., Kalman filter with a linear motion model) or feature-based tracking such as the mean-shift [Collins 2003] or KLT algorithm [Baker and Matthews 2004]. Our method, BRVO, is an online method that is based on a non-linear motion-model. BRVO is also an adaptive method that adjusts the parameters according to the environment. We briefly summarize the RVO motion model and discuss its advantages when handling sensor-based inputs.

## 3.1 Motion Prediction Methods using Agent-Based Motion Models

Our approach optimizes existing crowd simulation methods by integrating machine learning; it adapts the underlying crowd-simulation models based on what it learns from captured real-world crowd data. Some recent work has explored the use of crowd simulation as motion priors. Two notable examples of motion-prediction methods that use agent-based motion models are the previously mentioned approaches of LTA [Pellegrini et al. 2009] and ATTR [Yamaguchi et al. 2011]. We briefly discuss these two methods in order to highlight how our approach differs from them and to better understand the source of our performance improvement at predicting paths in real-world datasets (discussed in Section 5).

**Motion Model** LTA and ATTR use motion models based on an




(a) RVO State ($\mathbf{p}$, $\mathbf{v}$, $\mathbf{v}_{pref}$)

(b) Adaptive Refinement and Prediction

**Figure 1:** *(a)* ***RVO Simulation Overview*** *shows agent A's position ($\mathbf{p}$), preferred velocity ($\mathbf{v}_{pref}$) and actual velocity ($\mathbf{v}$), along with a neighboring agent B (both represented as circles). If the ORCA collision-avoidance constraints prevent an agent A from taking its preferred velocity, as shown here, the actual velocity will be the closest allowable velocity. Taken together, these elements form an agent's RVO state $\mathbf{x}$. (b)* ***Agent State Estimation*** *As new data is observed (blue dot) BRVO refines its estimate of a distribution of likely values of the RVO states (dashed ellipse). These parameters are then used with RVO to predict likely paths (as indicated by arrow).*

energy-minimization function. Their motion computation function takes into account various terms like steering to the destination, avoiding collisions with other agents, regulations in current speed, etc. ATTR also includes a parameter to capture the effect of group behaviors. Minimization of this motion function is used to compute the desired velocity.

**Offline Learning** Both these methods contain an offline preprocessing step that trains their motion models using real pedestrian video data, captured in the same scene in which the system is deployed. This offline learning is used to determine free parameters (such as the weight of various terms used to compute an agent's desired velocity) and for collision avoidance. In addition, ATTR learns about group behavior and destination locations, properties that affect individual behavior. The need for an offline learning phase reduces the potential applicability of these methods to mobile robots navigating in potentially unknown environments.

**Destination Prediction** LTA and ATTR both separate trajectory prediction from destination determination. In other words, destination information is not itself part of the prediction; instead, it must be given a priori in order to predict each pedestrian's motion. Both methods use offline, scene-specific processes to determine the destinations. In LTA, the destination is assumed to always lie on the opposite side of the scene, while ATTR uses an offline preprocessing step based on clustering to learn a fixed number of destinations within the scene.

Our method differs from LTA and ATTR in all three categories. First, our motion model is based on a velocity-space planning technique. Parameters and interaction information are used to compute geometric information in velocity space, rather than terms in an energy function. Second, our method learns parameters online rather than offline; this allows it to learn unique parameters for each individual, as well as to dynamically and adaptively refine each individual's parameters across time. Finally, our method learns the preferred velocity (medium-term, interim information) rather than the destination (long-term, final information) which is more appropriate given our online parameter-learning paradigm.

The flexible adaptation of preferred velocity offers many advantages over techniques that rely on global information. Global, constant parameters can give excellent results when, for example, the motion of a pedestrian is linear, or when its speed is constant over time. However, human motion changes dynamically, temporally and spatially. Adjusting to the changes dynamically, for example steering directions or speeds, reduces the deviation from the actual path, as compared to motion models that use constant parameters. We include a detailed discussion about the benefits of our method in Sec. 5.4.

### 3.2 RVO Multi-Agent Simulation

As part of our approach, we use an underlying multi-agent simulation technique to model individual goals and interactions between people. For this model, we chose a velocity-space reasoning technique based on Reciprocal Velocity Obstacles (RVO) [**?**]. RVO-based collision avoidance has previously been shown to reproduce important pedestrian behaviors such as lane formation, speed-density dependencies, and variations in human motion styles [Guy et al. 2010; Guy et al. 2011].

Our implementation of RVO is based on the publicly available RVO2-Library (`http://gamma.cs.unc.edu/RVO2`). This library implements an efficient variation of RVO that uses a set of linear constraints on an agent's velocities, known as *Optimal Reciprocal Collision Avoidance* (ORCA) constraints, in order to ensure that agents avoid collisions [van den Berg et al. 2011]. Each agent is assumed to have a position, a velocity, and a preferred velocity; if an agent's preferred velocity $\mathbf{v}_{pref}$ is forbidden by the ORCA constraints, that agent chooses the closest velocity which is not forbidden. Formally:

$$\mathbf{v}^{new} = \underset{\mathbf{v} \in ORCA}{\operatorname{argmin}} \|\mathbf{v} - \mathbf{v}_{pref}\|. \quad (1)$$

This process is illustrated in Fig. 1(a).

An agent's position is updated by integration over the new velocity $\mathbf{v}^{new}$ computed in Eqn. (1). An agent's preferred velocity is assumed to change slowly, and is modeled by a small amount of noise during each timestep. More details of the mathematical formulation are given in Sec 4. A thorough derivation of ORCA constraints is given in [van den Berg et al. 2011].

BRVO uses RVO combined with a filtering process. It can replace the entire framework of offline prediction methods: LTA or ATTR, for example, and the subsequent machine learning or preprocessing of data that they require. RVO was chosen as our motion model because it is especially suitable for sensor-based appli-

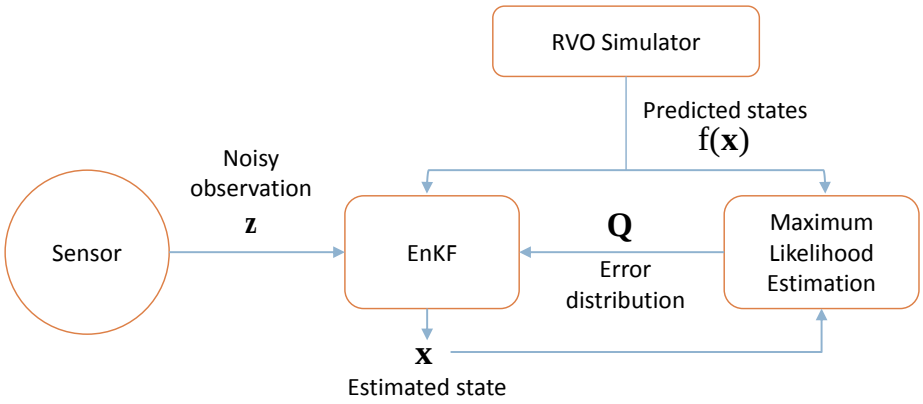



**Figure 2:** ***Overview of the Adaptive Motion Model.*** *We estimate current state* $\mathbf{x}$ *via an iterative process. Given noisy data observed by the sensor, RVO as a motion model, and the error distribution matrix* $\mathbf{Q}$*, we estimate the current state. The error distribution matrix* $\mathbf{Q}$ *is recomputed based on the difference between the observed state and the prediction* $f(\mathbf{x})$*, and is used to refine the current estimation of* $\mathbf{x}$*.*

cations. For example, as a velocity-based method, BRVO can handle a wide range of sampling rates and crowd densities. Force-based multi-agent simulation methods, like LTA and ATTR, compute energy potentials from the proximity of pedestrians and use these potentials as steering forces for their agents. These force-based methods thus require careful parameter tuning and small time steps to remain stable, and even with small simulation time steps, force-based methods are prone to generating oscillatory motions [Köster et al. 2013; Curtis et al. 2011]. These force-based methods are thus especially problematic when using video data, which has a sensing frequency (or frame rate) of 25 - 30 fps, or 0.033 to 0.04 seconds per frame; this high sampling rate must be reduced to decrease the computational burden for interactive applications, and force-based methods are unable to cope with the larger time steps created by reducing the sample rate. Velocity-based methods, however, remain stable even when sampling rates are decreased and relatively big time steps are used. RVO, which is a velocity-based method, is shown to be stable in large-time steps, as well as when simulating large, dense crowds[Curtis et al. 2012].

While we chose RVO for its stability in large time-steps and dense scenarios, the motion model part of BRVO is not specific to RVO. In other words, the motion computation formulation of LTA, ATTR, or any other agent-based motion model that uses preferred velocity or short-term destination information can be used instead of RVO. Our main goal is to demonstrate how an online filtering process can be combined with an RVO-based motion model to accurately predict pedestrian trajectories.

# 4 Bayesian-RVO

In this section, we provide the mathematical formulation of our Bayesian-RVO motion model, or **BRVO**. This model combines an Ensemble Kalman Filtering approach with the Expectation-Maximization (EM) algorithm to best estimate the approximate state of each agent, as well as the uncertainty in the model.

## 4.1 Problem Definition

In this section, we introduce our notation and the terminology used in the rest of the paper.

A *pedestrian* is assumed to be a human entity that shares the environment with a mobile robot. We treat pedestrians as autonomous entities that seek to avoid collisions with each other (but not necessarily with the robot). We use $n$ to denote the number of pedestrians in the environment. We assume that a robot's *sensor* produces a (noisy) estimate of the position of each pedestrian. Lastly, we assume that the robot has an estimate of impassable obstacles in the environment, represented as a series of connected line segments.

We define the computation of the BRVO motion model as a state-estimation problem. Formally, the problem we seek to solve is as follows. Given a set of observations, denoted as $\mathbf{z}_0 \cdots \mathbf{z}_k$, for each pedestrian at timestep $k$, we solve for the RVO state $\mathbf{x}_k$ that best reproduces the motion seen so far. Given this estimate, we can then use the RVO simulation model to determine the likely future path of each pedestrian.

We propose an iterative solution to this problem based on the EM-algorithm. Assume that a robot is operating under a sense-plan-act loop: the robot first measures new (noisy) positions of each pedestrian, then iteratively updates its estimate of each person's state. To do this, we run EnKF individually for each pedestrian. The joint state can be estimated using an RVO motion model. This model uses for its input the latest estimated positions of every pedestrian in the scene, and uses that input to compute the local collision avoidance of each pedestrian. Based on the prediction, the robot can create a local collision-free navigation plan for each pedestrian.

## 4.2 Model Overview

We use the RVO algorithm to represent the state of each sensed pedestrian in a robot's environment. The state of the pedestrian at timestep $k$ is represented as a 6D vector $\mathbf{x}_k$, which consists of an agent's 2D position, 2D velocity, and 2D preferred velocity. It is important to note that the preferred velocity is included as a part of the state vector and does not need to be manually pre-determined in a scene-specific fashion. We use a Bayesian inference process to adaptively find the state that best represents all previously observed motion data (accounting for sensing uncertainty) for each pedestrian in the robot's environment. In our case, we assume that the robot has an ability to observe the relative position of the pedestrians.

Given an agent's RVO state $\mathbf{x}_k$, we use the RVO collision-avoidance motion model, denoted here as $f$, to predict the agent's next state $\mathbf{x}_{k+1}$. We denote the state prediction error of $f$ at each time step as $\mathbf{q}$. This leads to our motion model of:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k) + \mathbf{q}. \tag{2}$$

Additionally, we assume that the sensing of the robot can be represented by a function $h$ that provides an observed state $\mathbf{z}_k$, which is a function of the person's true state $\hat{\mathbf{x}}_k$ plus some noise from the sensing processing, which is denoted as $\mathbf{r}$. That is:

$$\mathbf{z}_k = h(\hat{\mathbf{x}}_k) + \mathbf{r}. \tag{3}$$

An overview of this adaptive motion prediction model is given in Fig. 2.

**Simplifying Assumptions** The model given by Eqns. (2) and (3) is very general. In order to efficiently estimate the state $\mathbf{x}_k$ from the observations $\mathbf{z}_k$, we must make some simplifying assumptions, which allow us to develop a suitable learning approach for our adaptive model. First we assume that the error terms $q$ and $r$ are independent at each timestep, and that they follow a zero-meaned Gaussian distribution with covariances $Q$ and $R$, respectively. That is:

$$\mathbf{q} \sim N(0, Q), \tag{4}$$
$$\mathbf{r} \sim N(0, R). \tag{5}$$

We further assume that the sensor error $r$ is known or can be well-estimated. This is typically possible by making repeated measurements of known data points to compute the average error. In many cases, this error will be provided by the manufacturer of the sensing device. When using a tracking or recognition system to process the position input, $R$ should be estimated from the tracking and recognition error.

To summarize, the function $h$ is specified by the robot's sensors, and the matrix $R$ characterizes the estimated accuracy of these sensors. The $f$ function is the motion model that will be used to predict the motion of each agent (here RVO), and $Q$ represents the accuracy of this model.

Our BRVO formulation uses the RVO-based simulation model to represent the function $f$ and EnKF to estimate the simulation parameters that best fit the observed data. At a high level, EnKF operates by representing the potential state of an agent at each timestep as an ensemble (or collection) of discrete samples. Each sample is updated according to the motion model $f$. A mean and standard deviation of the samples is computed at every timestep in order to estimate the new state.

In addition, we adapt the EM-algorithm to estimate the model error $Q$ for each agent. Better estimation of $Q$ improves the Kalman filtering process, which in turn improves the predictions given by BRVO. This process is used iteratively to predict the next state and to refine the state estimation for each agent, as depicted in Fig 1(b). More specifically, we perform the EnKF and EM steps for each agent separately while simultaneously taking into account all agents present in the motion model $f(x)$. This gives more flexibility in dynamic scenes, allowing the method to handle cases when the robot moves or when pedestrians arrive or depart and change the number of agents in the sensed area. Because each inference step is performed per-agent on a fixed-size state (as opposed to a varying size state vector which considers all agents together), agents entering or leaving can be easily incorporated.

### 4.3 State Representation

The above representation can be more formally summarized with the following notation. We represent each agent's state, $\mathbf{x}$, as the six-dimensional set of RVO parameters (see Sec 3.2):

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}, \tag{6}$$

where $\mathbf{p}$ is the agent's position, $\mathbf{v}$ the velocity, and $\mathbf{v}_{pref}$ the preferred velocity. The crowd dynamics model $f$ is then:

$$f(\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}) = \begin{bmatrix} \mathbf{p} + \mathbf{v}\Delta t \\ \text{argmin}_{\mathbf{v} \in ORCA} \|\mathbf{v} - \mathbf{v}_{pref}\| \\ \mathbf{v}_{pref} \end{bmatrix}. \tag{7}$$

Also, the robot can observe the relative positions of the pedestrians, which can be represented as a function $h$ of the form:

$$h(\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}) = \mathbf{p}. \tag{8}$$

The EnKF use some assumptions for the observation model. In general though, our BRVO framework makes no assumption about the linearity of the sensing model. More complex sensing functions (for example, advanced computer vision techniques or the integration of multiple sensors) can be represented by modifying the function $h$ in accordance with the desired sensing model.

### 4.4 State Estimation

When $f$ and $h$ are linear functions, optimal estimation of the simulation state $\mathbf{x}_k$ is possible using Kalman Filtering. However, because $f$ corresponds to the RVO simulation model and we can use any arbitrary sensing model $h$, the resulting system is a non-linear system and there is no simple way to compute the optimal estimate.

Therefore, we use EnKF, a sampling-based extension of Kalman Filtering, to estimate the corresponding agent state for each pedestrian. EnKF is a Bayesian filtering process, which takes as input an estimate of the prediction error $\mathcal{Q}$ and observations $\mathbf{z}_0 \cdots \mathbf{z}_k$ and produces an estimate of the true pedestrian states $\mathbf{x}_k$ as a distribution of likely states $\mathcal{X}_k$. Given $n$ pedestrians to track, each with a $d$ dimensional state vector, the distributions $\mathcal{X}_k$ are represented in an ensemble representation where each distribution is represented by $m$ samples. EnKF provides an estimate of the true distribution of likely pedestrian states by representing it with these $m$ samples.

One benefit of performing EnKF for each pedestrian is that we can easily handle entering and leaving pedestrians, unlike a joint-state estimation. Instead, crowd interactions such as collision avoidance behavior is estimated by our motion model $f$, using the latest estimation of the pedestrians in the scene.

The ensemble representation for distributions is similar to a particle representation used in particle filters [Arulampalam et al. 2002], with the distinction that the underlying distribution is assumed to be Gaussian. The ensemble representation is widely used for estimation of systems with very large dimensional state spaces, such as climate forecasting [Hargreaves et al. 2004] and traffic conditions [Work et al. 2008]. In general, the EnKF algorithm works particularly well for high-dimensional state spaces [Evensen 2003] and can overcome some of the issues presented by non-linear dynamics [Blandin et al. 2012]. For a more detailed explanation of EnKF, we

(a) Campus (b) Bottleneck (c) Street (d) Students

**Figure 3:** ***Benchmarks used for our experiments*** *(a) In the Campus dataset, students walk past a fixed camera in uncrowded conditions. (b) In the Bottleneck dataset, multiple cameras track participants walking into a narrow hallway. (c) In the Street dataset, a fixed camera tracks pedestrian motion on a street. (d) In the Students dataset, a fixed camera tracks students' motion in a university.*

refer readers to a standard textbook in statistical inference such as [Casella and Berger 2002].

The motivation for this algorithmic choice is twofold. First, EnKF makes the same assumptions we laid forth in Sec 4.2: that is, a (potentially) non-linear $f$ and $h$ combined with a Gaussian representation of error. Second, as compared to methods such as particle filters, EnKF is very computationally efficient, providing higher accuracy for a given number of samples. This is an important consideration for the low-to-medium power onboard computers commonly found on a mobile robot, especially given a goal of online, real-time estimation.

### 4.5 Maximum Likelihood Estimation

The accuracy of the states estimated by the EnKF algorithm is a function of the parameters defined in Eqns 2-5: $f$, $Q$, $h$, and $R$. Three of these parameters are pre-determined given the problem set-up: the sensing function $h$ and the error distribution $R$ are determined by the sensor's specifications, and $f$ is determined by the motion model chosen. However, the choice of motion prediction error distribution $Q$ is still a free parameter. We propose a method of estimating the optimal value for $Q$ based on the Expectation Maximization or EM-algorithm.

The EM-algorithm is a generic framework for learning (locally) optimal parameters iteratively with latent variables. The algorithm alternates between two steps: an E-step, which computes expected values for various parameters, and an M-step, which computes the distribution maximizing the likelihood of the values computed during the E-step (for a more thorough discussion see [McLachlan and Krishnan 2008]).

In our case, the E-step is performed via the EnKF algorithm. This step estimates the most likely state given the parameters Q. For the M-step, we need to compute Q that maximizes the likelihood of values estimated from EnKF. This probability will be maximized with Q that best matches the observed error between the predicted state and the estimated state. We can compute this value simply by finding the average error for each sample in the ensemble at each timestep for each agent.

By iteratively performing the E and M steps, we continuously improve our estimate of Q, which will in turn improve the quality of the learning and the predictiveness of the method. Ideally, one should iterate over the E and M steps until the approach converges. In practice, the process converges fairly rapidly. Due to the online process nature of our approach, we limit the number of iterations to three, which we found to be empirically sufficient. We analyze the resulting improvement produced by the EM algorithm in Section 5.1.3.

### 4.6 Implementation

Pseudocode for our BRVO algorithm is given in Algorithm 1. We represent the distribution of likely RVO states as an ensemble of $m$ samples. We set $m = 1000$ for the results shown in Section 5. Lines 2 through 6 perform a stochastic prediction of the likely next states. Lines 7 through 10 correct this predicted value based on the observed data from the robot's sensor. $Z_k$ is a measurement error covariance matrix, and $\Sigma_k Z_k^{-1}$ is an ensemble version of Kalman Gain matrix. Lines 11 through 14 perform a maximum likelihood estimation of the uncertainty in the prediction.

**Algorithm 1:** Bayesian-RVO

**Input**: Observed positions over time $\mathbf{z}_1 ... \mathbf{z}_t$, crowd motion simulator $f$, estimated initial error covariance $\mathbf{Q}$, sensor function $h$, sensor noise $\mathbf{R}$, and the number of samples $m$.
**Output**: Estimated agent's state distributions $\mathbf{x}_1 ... \mathbf{x}_t$

1 **foreach** $k \in 1 \dots t$ **do**
  `// EnKF Predict Step`
2  **foreach** $i \in 1 \dots m$ **do**
3  Draw $\mathbf{q}_{k-1}^{(i)}$ from $\mathbf{Q}$, $\hat{\mathbf{x}}_k^{(i)} = f(\hat{\mathbf{x}}_{k-1}^{(i)}) + \mathbf{q}_{k-1}^{(i)}$ ;
4  Draw $\mathbf{r}_k^{(i)}$ from $\mathbf{R}$, $\hat{\mathbf{z}}_k^{(i)} = h(\hat{\mathbf{x}}_k^{(i)}) + \mathbf{r}_k^{(i)}$;
5  $\bar{\mathbf{z}}_k = \frac{1}{m} \sum_{i=1}^{m} \hat{\mathbf{z}}_k^{(i)}$;
6  $Z_k = \frac{1}{m} \sum_{i=1}^{m} (\hat{\mathbf{z}}_k^{(i)} - \bar{\mathbf{z}}_k)(\hat{\mathbf{z}}_k^{(i)} - \bar{\mathbf{z}}_k)^T$;
  `// EnKF Correct Step`
7  $\bar{\mathbf{x}}_k = \frac{1}{m} \sum_{i=1}^{m} \hat{\mathbf{x}}_k^{(i)}$;
8  $\Sigma_k = \frac{1}{m} \sum_{i=1}^{m} (\hat{\mathbf{x}}_k^{(i)} - \bar{\mathbf{x}}_k)(\hat{\mathbf{z}}_k^{(i)} - \bar{\mathbf{z}}_k)^T$;
9  **foreach** $i \in 1 \dots m$ **do**
10  $\hat{\mathbf{x}}_k^{(i)} = \hat{\mathbf{x}}_k^{(i)} + \Sigma_k Z_k^{-1} (\mathbf{z}_k - \hat{\mathbf{z}}_k^{(i)})$;
  `// Maximum Likelihood Estimation`
11  $\mathbf{Q}_{k-1} = \mathbf{Q}$;
12  **foreach** $i \in 1 \dots m$ **do**
13  $\mathbf{Q}_k += (\hat{\mathbf{x}}_k^{(i)} - f(\hat{\mathbf{x}}_{k-1}^{(i)}))(\hat{\mathbf{x}}_k^{(i)} - f(\hat{\mathbf{x}}_{k-1}^{(i)}))^T$;
14  $\mathbf{Q} = \frac{k-1}{k} \mathbf{Q}_{k-1} + \frac{1}{k} \mathbf{Q}_k$

## 5 Results and Analysis

In this section, we show the results from our BRVO approach when applied to real-world datasets. We begin by isolating the effect of each component of BRVO separately, namely: EnKF, EM and individual parameter learning (Section 5.1). We then measure the robustness of our approach across a variety of factors commonly

found in real world situations including: sensor noise, low sampling rates, and spatially varying densities. Finally, we provide a quantitative comparison to other recent techniques across a variety of datasets.

Our analysis here is performed on four different datasets with various levels of noise and different sampling rates, illustrated in Fig. 3. We briefly describe each dataset below:

**Campus** Video data of students on campus recorded from the top of the ETH main building in Zurich, extracted manually every 0.4 second [Pellegrini et al. 2009]. We extracted three sequences from this data, each containing about 10 seconds of pedestrian interaction: Campus-1 (7 pedestrians), Campus-2 (11 pedestrians), Campus-3 (18 pedestrians).

**Bottleneck** Optical tracking equipment captured the motion of participants in a lab environment [Boltes et al. 2010]. Participants moved through a bottleneck opening into a narrow passage. Data was collected with a passage width of 2.5 meters. The study has about 170 participants, and we use the trajectory data subsampled at a rate of 0.4 second.

**Street** This video of low-density pedestrian traffic was recorded from a street view [Lerner et al. 2009]. Manually extracted pedestrian trajectories are provided with the video. The dataset contains two sequences. Street-1 contains motion of 148 pedestrians over a period of roughly 5 minutes, and Street-2 contains the motion of 204 pedestrians over a period of roughly 7 minutes.

**Students** This video was recorded from a street view [Lerner et al. 2009]. The dataset contains the motion of 434 students over a period of roughly 3 minutes, tracked manually.

To explore the relative performance of our method we will compare it to other typical prediction approaches commonly found in robotics applications as described below:

**Constant Velocity (ConstVel)** This model assumes that all agents will continue their most recent instantaneous velocity (inferred from the last two positions) for the immediate future (Eqn 9).

$$f(\begin{bmatrix}\mathbf{p}\\ \mathbf{v}\end{bmatrix}) = \begin{bmatrix}\mathbf{p}+\mathbf{v}\Delta t\\ \mathbf{v_{k-1}}\end{bmatrix} = \begin{bmatrix}\mathbf{p}+\mathbf{v}\Delta t\\ \frac{\mathbf{p_{k-1}}-\mathbf{p_{k-2}}}{\mathbf{\Delta t}}\end{bmatrix}. \quad (9)$$

**Constant Acceleration (ConstAcc)** This model assumes that all agents will continue their most recent acceleration (inferred from the last two velocities) for the immediate future (Eqn 10).

$$f(\begin{bmatrix}\mathbf{p}\\ \mathbf{v}\\ \mathbf{a}\end{bmatrix}) = \begin{bmatrix}\mathbf{p}+\mathbf{v}\Delta t\\ \mathbf{v}+\mathbf{a}\Delta t\\ \mathbf{a_{k-1}}\end{bmatrix} = \begin{bmatrix}\mathbf{p}+\mathbf{v}\Delta t\\ \mathbf{v}+\mathbf{a}\Delta t\\ \frac{\mathbf{v_{k-1}}-\mathbf{v_{k-2}}}{\mathbf{\Delta t}}\end{bmatrix}. \quad (10)$$

**Kalman Filtering (KF)** We use a position and velocity based Kalman Filter (i.e., using Eqn 9 as the model) which uses standard Gaussian filtering techniques to account for potential noise in the environment.

Unlike the ConstVel and ConstAcc models, using a Kalman filter will account for noise in the sensing and motion models. However, this comes at the cost of a delayed response to changes in the environment (due to data smoothing) leading to KF performing worse than ConstVel in situations where there is little noise and smooth, consistent motion. BRVO avoids this issue as demonstrated in Section 5.2.

We also compare the results against two recently proposed *offline* pedestrian tracking methods: LTA [Pellegrini et al. 2009] and ATTR [Yamaguchi et al. 2011]. Both methods use offline training to learn models of typical pedestrian goals and speeds for a given observation area. Because of this training step, both methods are inappropriate for use in a mobile robot. However, they provide a good understanding of the state-of-the-art in pedestrian tracking. Surprisingly, in all of the above datasets, BRVO outperforms even these offline model suggesting the flexibility of learning a model per-agent outweighs the advantages from offline training. More discussion of this comparison is given in Section 5.3.

In all cases, we report errors as the distance between ground-truth (manually annotated) positions and the estimated positions. We also provide various quantitative and qualitative performance measures including the long-term prediction success rate, shapes of the trajectories and prediction error distributions.

## 5.1 Method Analysis

BRVO has three main components: a probabilistic Bayesian inference framework, per-agent online state estimation, and a continuing refinement of filtering parameters through the EM algorithm. Each of these components provides a substantial contribution to the overall performance.

### 5.1.1 EnKF for Online Learning

Different approaches to Bayesian inference can be used to perform the online learning task. Here we compare EnKF, Particle filter, and Unscented Kalman filter [Wan and Van der Merwe 2000], all using RVO as the motion model, all without EM. For this experiment, we use the three sequences from various points in the the Campus dataset.

The first five frames of ground truth positions are used as sensor input $\mathbf{z}$, and each method is issued to perform prediction for the remaining frames with the learned parameters. We use a fixed $\mathbf{Q}$ for each learning and prediction step: $[0.5m] * I$. The number of samples for both EnKF and Particle filter is 50000. We measured the mean error (squared distance) between the predicted position and ground truth position. The results are summarized in Table 1.

| | Campus-1 | Campus-2 | Campus-3 |
|---|---|---|---|
| EnKF | **0.359** | **0.462** | **0.388** |
| Particle filter | 0.366 | 0**.462** | 0.389 |
| Unscented KF | 0.476 | 0.501 | 0.420 |

**Table 1:** ***Comparisons of different Bayesian learning algorithms*** *Root mean squared error between the predicted position and ground truth position in meters (best shown in bold).*

Both EnKF and Particle Filter outperform Unscented Kalman Filter, with EnKF having the best average performance. While other sampling based approaches such as Particle filters may provide similar results, EnKF's consistent performance across these and other datasets motivated our choice of EnKF for Bayesian inference throughout this work.

### 5.1.2 Online vs Offline Learning

We can measure the improvements made by introducing online parameter-estimation technique, by comparing our method to a closely related offline approach. Following the approach proposed in LTA [Pellegrini et al. 2009] and ATTR [Yamaguchi et al. 2011], we developed an offline prediction technique based on RVO referred to here as **RVO-offline**. At a high level, RVO-offline uses a genetic algorithm to optimize RVO parameters to best match the

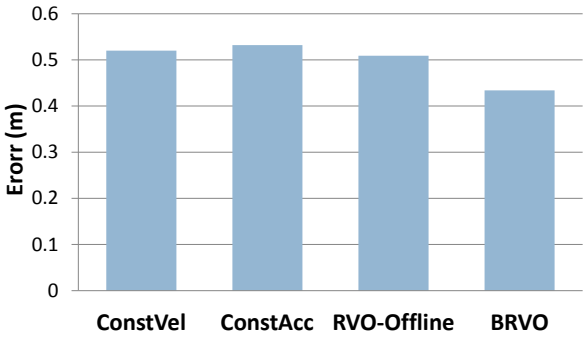



**Figure 4:** ***Comparison of Error on Campus Dataset.*** *BRVO produces less error than Constant Velocity and Constant Acceleration predictions. Additionally, BRVO results in less error than using a static version of RVO whose parameters were fitted to the data using an offline global optimization.*

training dataset of the ground truth motion of individuals in the environment. The destination of each agent is set side of the environment opposite of where the agent entered (as done in [Pellegrini et al. 2009]). For ConstVel, ConstAcc, and BRVO, we assume that we get good sensor inputs for first two frames and compute the initial velocity for each pedestrian from the first two groundtruth positions. For RVO-Offline, we assume that we have good global information about destinations and motion parameters including preferred speed, and compute the initial velocity from these parameters.

We compare this RVO-offline model with the constant velocity model, the constant acceleration model, and with BRVO using the above mentioned sequences from the Campus dataset. To compute the error, we use the first half of the data set for training and the second half for testing. Each method attempts to predict the final agent position from the position at the halfway point. Fig. 4 shows the mean error for different approaches.

While the static, offline-learning RVO model performs slightly better than the simpler motion models, BRVO offers an even more significant reduction in error, demonstrating the advantage of learning individual time-varying parameters for each agent. Additionally, unlike offline models, BRVO does not need prior knowledge of the current environment (such as likely goals), instead learning each agents destination dynamically through EnKF with RVO. The difficultly in choosing an individual's goal position, is one of the main challenges to using offline methods in an autonomous robot system. More discussion of the impact of goals in offline methods is given in Section 5.3

#### 5.1.3 EM based estimation refinement

As discussed in Section 4.5, our proposed BRVO approach uses the EM-algorithm to automatically refine the parameters used in the online Bayesian estimation. The effect of EM can be demonstrated visually by graphing the estimated uncertainty in state as shown in Fig. 5(a); as new data is presented and the pedestrians interact with each other, the estimated uncertainty, $\mathbf{Q}$, decreases. Without the EM step, the initially given estimated uncertainty $\mathbf{Q}$ must be used without refining its values.

Figure 5(b) shows the quantitative advantage of using EM, by comparing our results in the Campus sequences with EM to those without. When using high quality data with little noise, EM shows a slight improvement in the results. If we artificially add noise to the data (while keeping all other parameters constant), the effect of the active feedback loop becomes even more important and the EM algorithm offers increasing improvements in error estimation.

### 5.2 Environmental Variations

Many factors in a robot's environment can impact how well a predictive tracking method such as BRVO works. For example, poor lighting conditions can reduce the efficacy of vision based sensors increasing the noise in the sensor data. Likewise, in dense scenarios, people can tend to change their velocities frequently to avoid collisions creating increasing the uncertainty in their motion. Similarly, when a robot's energy gets low it may choose to reduce the frequency of sampling sensor data to save power. In all of these scenarios, the difficulty in predicting a pedestrians motion can increase significantly. Importantly, the advantage that BRVO has over simple prediction methods grows larger as scenarios grow in complexity through additional sensor noise, more dense environments, or less frequent data readings as shown in the following experiments.

#### 5.2.1 Noisy Data

To analyze how BRVO responds to noise in the data, we compare its performance across varying levels of synthetic noise added to the ground truth data in the Campus dataset. Like before, BRVO learns for the first 5 frames, and uses that information to predict the position for remaining frames.

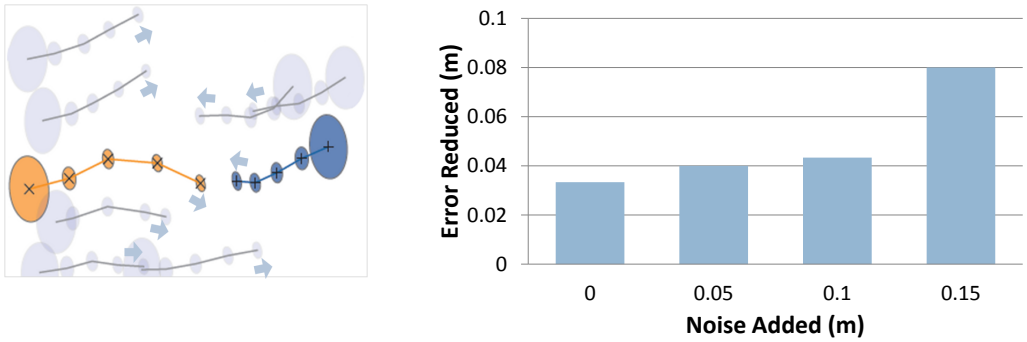
Fig. 6 compares predictions from BRVO, constant velocity, and constant acceleration for one pedestrian's moving right-to-left across the campus. The figure shows the prediction given with varying amounts of noise levels added to the training data (respectively, 0.05m noise, 0.1m noise and 0.15m noise). After the training frames, we assume that no further sensor information is given and predict the paths using only the motion models. As can be seen in the figure, BRVO reduces sensitivity to this type of noise and performs better overall at predicting both the direction of the path and the absolute speed of the pedestrian.

A comparison of the average prediction error across all of the Campus sequences is given in Fig. 7 which compares BRVO with the constant velocity model, constant acceleration model, and KF. Both KF and BRVO are initialized with the same conditions (e.g., Sensor noise and initial error covariance matrix). While adding noise always increases the prediction error, BRVO was less sensitive to the noise than other methods. Fig. 8 shows the percentage of correctly predicted paths within varying accuracy thresholds. At an accuracy threshold of 0.5m, BRVO far outperforms the constant velocity and constant acceleration models (44% vs 8% and 11% respectively) even with little noise. With larger amounts of noise, these differences tend to increase.

#### 5.2.2 Density Dependance

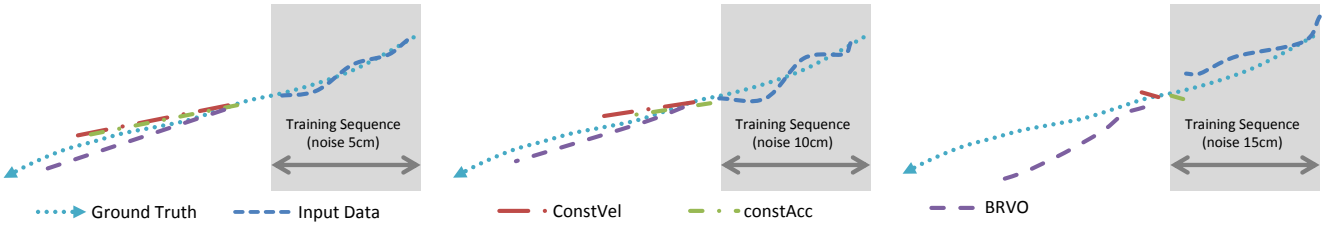
In a dense crowd, an individual's motion is restricted significantly by his or her neighbors. It is thus difficult to predict pedestrian trajectories without taking into account the neighbors. Because individuals in the bottleneck scenario undergo many different densities, it provides the clearest effect of density has on prediction accuracy. For analysis purposes, the scenario was spilt in three sections: a dense front section where people backup trying to make their way through the narrow bottleneck opening ( 3.6 $people/m^2$), a more moderately dense section at the entrance (3 $people/m^2$), and the hallway itself which is less dense (2 $people/m^2$). Note that this last region is not only the least dense, but has the simplest motion consisting primarily of moving straight through the hallway.

Fig. 9 compares BRVO's error to that of the constant velocity, con-



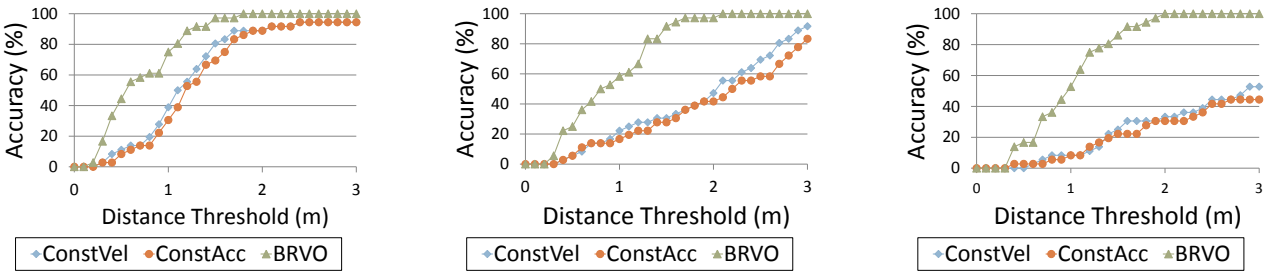

(a) Estimation Refinement (b) Improvement from EM feedback Loop

**Figure 5:** ***The effect of the EM algorithm.*** *(a) This figure shows the trajectory of each agent and the estimated error distribution (ellipse) for the first five frames of Campus-3 data. The estimated error distributions gradually reduces as agents interact. For two highlighted agents (orange and blue), their estimated positions are marked with 'X'. (b) The improvement provided by the EM feedback loop for various amounts of noise. As the noise increases, this feedback becomes more important.*




(a) 0.05m (b) 0.1m (c) 0.15m

**Figure 6:** ***Path Comparisons*** *These diagrams shows the paths predicted using BRVO, Constant Velocity, and Constant Acceleration. For each case, ground-truth data with 0.05m noise, 0.1m noise and 0.15m are given as input for learning (blue dotted lines in shaded regions). BRVO produces better predictions even with a large amount of noise.*




(a) 0.05m (b) 0.1m (c) 0.15m

**Figure 8:** ***Prediction Accuracy (higher is better)*** *(a-c) Shows prediction accuracy across various accuracy thresholds. The analysis is repeated at three noise levels. For all accuracy thresholds and for all noise levels BRVO produces more accurate predictions than the Constant Velocity or Constant Acceleration models. The advantage is most significant for large amounts of noise in the sensor data, as in (c).*

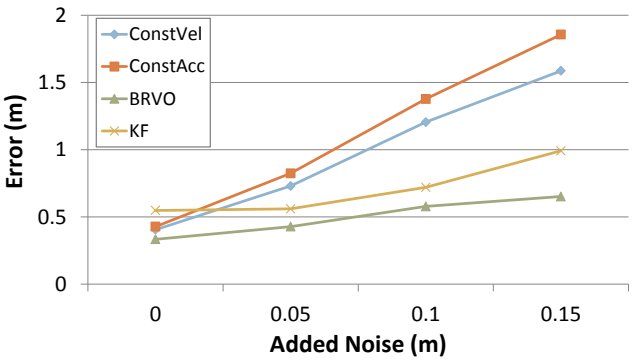



**Figure 7:** ***Mean prediction error (lower is better).*** *Prediction error after 7 frames (2.8s) on Campus dataset. As compared to the Constant Velocity and Constant Acceleration models, BRVO can better cope with varying amounts of noises in the data.*

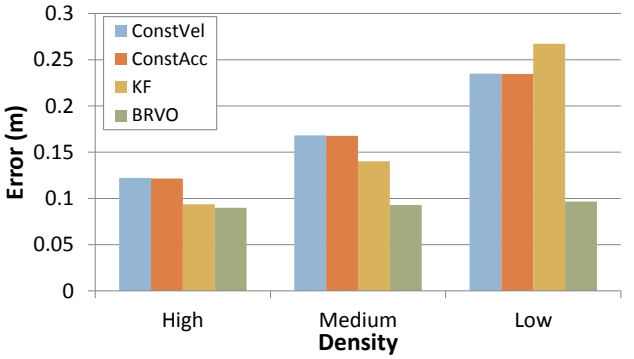



**Figure 9:** ***Error at Various Densities (lower is better).*** *In high-density scenarios, the agents move relatively slowly, and even simple models such as constant velocity perform about as well as BRVO. However, in low- and medium-density scenarios, where the pedestrians tend to move faster, BRVO provides more accurate motion prediction than other models. KF fails to quickly adjust to rapid speed changes in the low-density region, resulting in larger errors than other methods. In general, BRVO performs consistently well across various densities.*

stant acceleration, and Kalman filter models. The constant velocity, constant acceleration, and Kalman filter models have large variations in error for different regions of the scenario with different densities. The error is smallest in the highest density region. This is because the speed of the pedestrians are relatively slow, and the effect of the wrong prediction does not produce a large deviation from the ground-truth. As the density gets lower and the speed increases, we observe that the error increases. Unlike constant velocity and constant acceleration, which instantly adopts the speed, Kalman filter model have latency adapting to the abrupt speed changes, which may have resulted in a bigger error in the lower density region. In contrast, the BRVO approach performs well across all densities because it can dynamically adapt the parameters for each agent for each frame.

#### 5.2.3 Sampling Rate Variations

Our method also works well with very large timesteps, i.e., when sensor data is collected at a sparse rate over long intervals. To demonstrate this, we show the results on the Street dataset with varying sampling intervals to sub-sample the data. We chose Street-1 scenario, which has the highest framerate (at 0.04s per frame) of all the datasets, and then sub-sampled the data to reduce the effective framerate. Fig. 10 shows the graph of the mean error versus the sampling interval. The results show that our method performs very well compared to the constant velocity and constant acceleration model across all the sampling intervals, and has less sensitivity to low framerates.

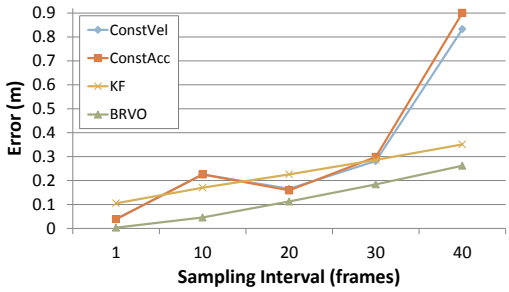



**Figure 10:** ***Error vs Sampling Interval*** *As the sampling interval increases, the error of Constant Velocity and Constant Acceleration estimations grows much larger than that of KF or BRVO. BRVO results have the lowest error among the four methods.*

### 5.3 Model Performance Comparison

Taken together, the results provided in this section demonstrate the ability of BRVO to provide robust, high-quality motion prediction in a variety of difficult scenarios. We can directly compare our results with the results of LTA [Pellegrini et al. 2009] and ATTR [Yamaguchi et al. 2011], which report performance numbers for some of the same benchmarks. Because LTA and ATTR require offline training, we also measure the performance of KF alone in the same scenarios, in order to provide a comparison with an online method, but with a simple linear motion model.

We use the Street-1, Street-2 and Students datasets, all sampled every 1.6 seconds, and measure mean prediction error for every agent in the scene during the entire video sequence. Fig. 11 compares the prediction accuracy of LTA, ATTR in various configurations, KF, and BRVO with LIN (the constant velocity model). Our method outperforms LTA and ATTR with 18-40% error reduction rate in across the three different scenarios. LTA and ATTR use the ground truth destinations for prediction; LTA+D and ATTR+DG use destinations learned offline, as explained in [Yamaguchi et al. 2011]; ATTR+DG uses grouping information learned offline. Even though BRVO is an online method, it shows significant improvement in prediction accuracy on all three datasets, producing less error than other approaches.

We observe a relatively large error when using the students dataset. This dataset is especially difficult to estimate, as explained in [Yamaguchi et al. 2011]; the irregular behavior of the pedestrians, including sudden stops, wandering, and chatting, reduce the predictive performance. This reduced predictability also affects the KF test, since it fails to quickly adjust to the irregular behaviors.

### 5.4 Analysis

The ability to learn the preferred velocity on the fly comes from the BRVO framework. BRVO can be compared with other prediction methods that use motion models followed by data pre-processing and offline parameter learning (e.g., LTA or ATTR). We have shown quantitative comparisons with LTA and ATTR on three different

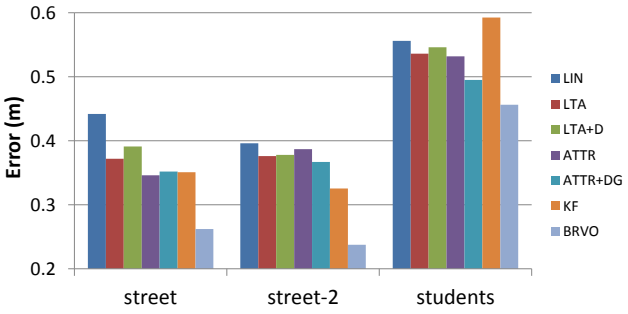



**Figure 11:** ***Comparison to State-of-the-art Offline Methods*** *We compare the average error of LIN (linear velocity), LTA, ATTR, KF and BRVO. Our method outperforms LTA and ATTR, with an 18-40% error reduction rate over LIN in all three different scenarios. Significant improvement is made; compare the 4-16% and 7-20% error reduction rate of LTA and ATTR over LIN, respectively.*

datasets. Overall, the differences make our approach better suited to the domain of mobile robot navigation.

We can also estimate how much of BRVO's performance improvement comes from the addition of the filtering process and how much from using the RVO motion model, although we believe that combining the motion model and filtering algorithm results in considerable improvement. To quantify the contributions of each, we provided additional comparisons: one with the constant-velocity model and one with the Kalman filter and the constant-velocity model. These two models are both online methods that use a linear motion model. The Kalman filter method is more resilient with the noisy inputs than is the constant velocity model, but adjusts more slowly when inputs change rapidly. Thus, the performance is influenced by the characteristics of the data. We have shown that our method actually outperforms both the constant-velocity and Kalman-filter methods in various scenarios.

## 6 Robot Navigation with BRVO

One potential application of our BRVO is for safer navigation for autonomous robot vehicles through areas of dense pedestrian traffic. Here we describe a simple method to integrate BRVO with the GVO (Generalized Velocity Obstacle) motion-planning algorithm proposed in [Wilkie et al. 2009] to achieve more effective navigation of a simulated car-like robot through a busy walkway. The GVO navigation method is a velocity obstacle based technique for navigating robots with kinematic constraints. In our case, we use car-like kinematic constraints, and we assume the robot has the ability to sense the positions of nearby moving obstacles (such as pedestrians) though with some noise. The robot uses our BRVO technique to predict the motion of each pedestrian as it navigates through the crowded walkway to its goal position.

The robot's configuration is represented as its position $(x, y)$ and the orientation $\phi$. The robot has controls $u_s$ and $u_\phi$, which are the speed and steering angle of the robot, respectively. Fig. 12 shows the kinematic model of the robot. Its constraints are defined as follows:

$$\begin{aligned} x'(t) &= u_s \cos\theta(t), \\ y'(t) &= u_s \sin\theta(t), \\ \theta'(t) &= u_s \frac{\tan u_\phi}{L}, \end{aligned} \tag{11}$$

where $L$ is the wheelbase of the robot. We assume $L = 1$m.

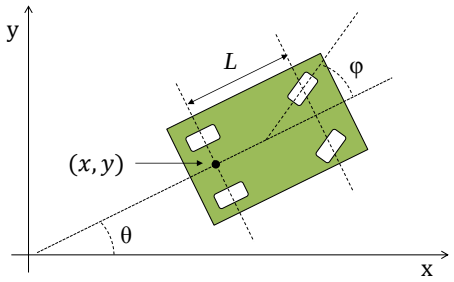



**Figure 12:** ***Kinematic model of the robot*** *The robot is modeled as a simple car at position $(x, y)$ and with orientation $\theta$. $\phi$ is a steering angle and L is a wheelbase. Our robot has the wheelbase $L = 1m$.*

Assuming that the control remains constant for the time interval, the robot's position $R(t, u)$ at time $t$ given the control $u$ can be derived as follows:

$$R(t,u) = \begin{pmatrix} \frac{1}{\tan(u_\phi)} \sin(u_s \tan(u_\phi)t) \\ -\frac{1}{\tan(u_\phi)} \cos(u_s \tan(u_\phi)t) + \frac{1}{\tan(u_\phi)} \end{pmatrix}. \tag{12}$$

GVO samples the space of controls for a robot to determine which controls lead to collisions with obstacles. This is done using an algebraic formulation for the minimum distance between the robot, given its kinematic model, and an obstacle, given its predicted trajectory, up to a time horizon. For our work, the minimum distance between a robot using the simple-car kinematic model and a linearly-moving obstacle is derived. We limit the maximum velocity of the robot to $1.5m/s$. For each control sample, this minimum distance is solved numerically. Every control that yields a distance greater than the sum of the robot and obstacle widths is considered free. Of these free controls, a control is then selected that brings the robot closest to its goal. Formally, the actual control $u$ is:

$$u = \underset{u' \notin VO}{\operatorname{argmin}}\, dist(R(t, u'), g, t')), \tag{13}$$

where $u'$ is subject to a velocity obstacle constraint for each moving obstacle, and $VO$ is the velocity obstacle of the robot. $dist(R(t, u'), g, t')$ is the minimum distance between the robot and the robot's goal position $g$, over a time horizon $t'$. We use 5 seconds for $t'$, which is bigger than the simulation update time horizon to give some a 'look ahead' effect. The computation of $u$ is a slight modification from the original formulation [Wilkie et al. 2009], where $u$ is selected from $u'$ which is closest to the optimal control $u*$. Instead of selecting the closest control, we choose the control which actually brings the robot closest to the goal.

When navigating, the robot uses BRVO to predict upcoming pedestrian motion and must avoid any steering inputs which (as determined by Equation 12) will collide with the predicted pedestrian positions. In essence, we assume a non-cooperative environment, where the pedestrians may not actively avoid collisions with the robot and the robot must assume 100% responsibility for collision avoidance (i.e., asymmetric behavior).

We use the Campus-1 (7 pedestrians), Campus-2 (11 pedestrians), and Campus-3 (18 pedestrians) data sequences from the **Campus** dataset (See Fig.3 (a)) to measure the performance of the robot navigation. The robot is given an initial position at one side of the walkway and is asked to move through the pedestrians to the opposite side. Given the importance of safety in pedestrian settings, if at any point the robot fails to find collision free trajectory which moves forward along the path, the robot will stop or turn back towards

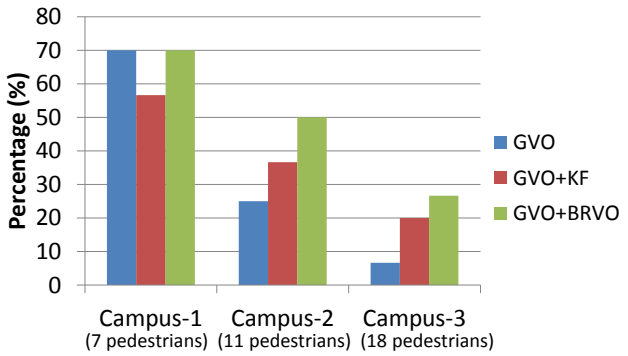



**Figure 13:** ***Performance of the robot navigation*** *We measured the percentage of the trajectories in which the robot reached farther than halfway to the goal position without any collision during the data sequence, using only GVO (blue bars), GVO with KF (red bars), and GVO with BRVO (green bars), both with 15cm sensor noise. In many of these scenarios, using GVO for navigation often caused a robot to stop moving through the crowd to avoid collisions (the freezing robot problem). GVO-KF shows lower performance in very sparse scenario, but outperforms GVO-only method as the number of pedestrians increases. GVO-BRVO algorithm further improves the navigation, especially for more challenging scenarios with more pedestrians.*

the start. Given this setup, we evaluate the percentage of times the robot is able to make it more than halfway through the pedestrian crossing without colliding with a pedestrian, or needed to stop or turn back.

We run experiments on all the three sequences, assuming a sensing error of $\pm$15cm noise in positional estimates, and a sampling rate of 2.5Hz to allow adequate time for any visual processing needed to detect pedestrians in the sensed area. We collect the mean of 30 runs for each sequence; the robot's initial position and goal position are randomly chosen for each run.

We compare the robot navigations in various combinations: GVO alone (using a constant-velocity model), GVO with GVO-KF, and GVO with BRVO. Fig. 13 shows the results with GVO only (blue bars), GVO-KF (red bars), and GVO-BRVO (green bars). As the scenarios get denser, the robots navigating with GVO alone tended to avoid collisions by staying still, displaying the same freezing robot problem as discussed by other researchers (see for example [Trautman and Krause 2010]). In very sparse crowds, GVO + a linear method performs slightly better than GVO-KF, but the performance rapidly drops as the number of pedestrians increases. The GVO-BRVO algorithm improves the navigation more than GVO-KF, especially for more challenging scenarios. Compared to GVO-alone, GVO-BRVO more than doubled the task-completion rate.

We also performed the same experiments using the ground-truth data without any sensor noise added. In this case, GVO-BRVO achieved 23% and 26% better performance over GVO-only and GVO-KF, respectively.

Similar results were seen in other datasets. Fig.3 shows an example path of the robot navigating through pedestrian motion taken from the **Students** dataset (with a pedestrian density of .35 $ppl/m^2$). The result shown comes from an experimental setup with an even lower sampling rate (.6Hz) and using only 100 ensemble samples in EnKF. We believe that this low sampling rate, and low processing requirement, may be appropriate for many types of low-powered mobile robots. Even with these reduced computational requirements, we still achieved about a 50% task completion rate.

This series of experiments indicates that BRVO can improve planning in uncertain environments: environments with dynamic obstacles or with sensor limitations, including noisy or sparse sensor inputs. Though better prediction does not guarantee better navigation, and the freezing robot problem can still occur [Trautman and Krause 2010], we believe that better prediction algorithms can improve the robotic navigation.

We also believe that BRVO can be combined with other robot navigation methods in a cooperative setup, since it improves the performance of RVO-based motion models in noisy-data situations, as discussed in [Trautman et al. 2013]. More importantly, BRVO does not need prior knowledge about the scene, such as destinations or goal positions and motion priors. BRVO-GVO, however, requires static obstacles to compute robot navigation. Motion prediction using BRVO does not require knowledge about static obstacles, although static obstacles that strongly constrain the motion of the pedestrians may improve its accuracy (e.g., **Bottleneck** dataset). Instead of the **Bottleneck** dataset, our experiments are performed without specifying the static obstacles. These features of BRVO can be a considerable benefit for navigating robots or autonomous wheelchairs in real-world scenes.

## 7 Conclusion and Future Work

We have presented a method to predict pedestrian trajectories using agent-based velocity-space reasoning. The BRVO model that we have introduced is an online motion-prediction method that learns per-agent motion models. Even without prior knowledge about the environment, it performs better than offline approaches that do have prior knowledge. We have demonstrated the benefits of our approach on several datasets, showing our model's performance with varying amounts of sensor noise, interaction levels, and densities. Specifically, we have shown that our approach performs very well with noisy data and in low framerate scenarios. We also highlight BRVO's performance in dynamic scenarios with density (spatial) and speed (temporal) variation.

BRVO assumes no prior knowledge of the scenario which it will navigate; it is thus uniquely well-suited for mobile robots that may frequently encounter new obstacles and that must handle pedestrians entering and leaving the environment. We have shown that by learning an individualized motion model for each observed pedestrian, our online motion-prediction model can perform better than less responsive offline motion models. We also showed that our method can be integrated with recent local navigation techniques to improve task completion rates and reduce instances of the freezing robot problem.

BRVO has also been used for pedestrian tracking applications and shown to increase tracking performance in various scenarios [Bera and Manocha 2014; Liu et al. 2014]. In the future, we would like to explore a method for local, dynamic group behavior estimation, to further improve the performance of BRVO.

## Acknowledgment

We would like to thank Jur van den Berg for his help. This research is supported in part by ARO Contract W911NF-10-1-0506, NSF awards 0917040, 0904990, 100057 and 1117127, and Intel.

## References

ARULAMPALAM, M., MASKELL, S., GORDON, N., AND CLAPP, T. 2002. A tutorial on particle filters for online nonlinear/non-

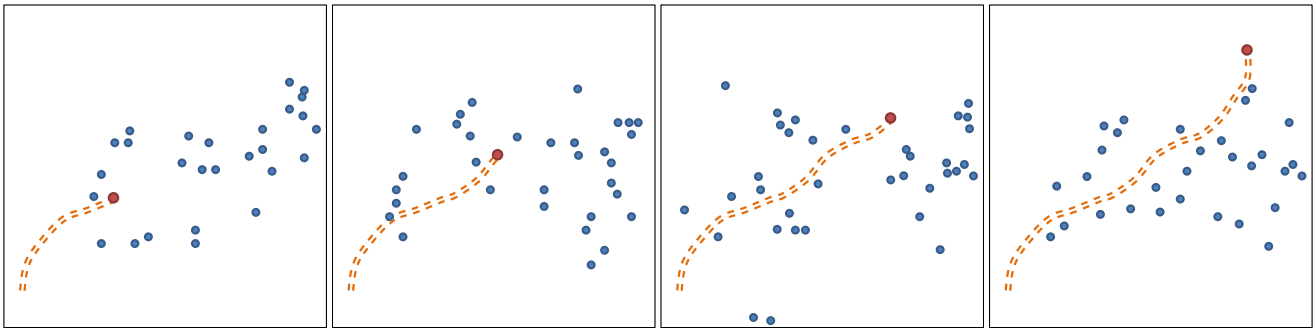


**Figure 14:** ***Example robot trajectory*** *navigating through the crowd in* ***Students*** *dataset. Blue circles represent current pedestrian positions, red circles are the current position of the robot, and orange dotted lines are the previous positions of the robot.*


gaussian bayesian tracking. *IEEE Transactions on Signal Processing 50*, 2, 174–188.

BAKER, S., AND MATTHEWS, I. 2004. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vision 56*, 3 (Feb.), 221–255.

BENNEWITZ, M., BURGARD, W., CIELNIAK, G., AND THRUN, S. 2005. Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research 24*, 1, 31–48.

BERA, A., AND MANOCHA, D. 2014. Realtime multilevel crowd tracking using reciprocal velocity obstacles. *IEEE International Conference on Pattern Recognition* (Feb).

BLANDIN, S., COUQUE, A., BAYEN, A., AND WORK, D. 2012. On sequential data assimilation for scalar macroscopic traffic flow models. *Physica D: Nonlinear Phenomena*.

BOLTES, M., SEYFRIED, A., STEFFEN, B., AND SCHADSCHNEIDER, A. 2010. Automatic extraction of pedestrian trajectories from video recordings. *Pedestrian and Evacuation Dynamics 2008*, 43–54.

BRUCE, A., AND GORDON, G. 2004. Better motion prediction for people-tracking. In *Proc. of the International Conference on Robotics and Automation (ICRA), New Orleans, USA*.

CASELLA, G., AND BERGER, R. 2002. *Statistical inference*. Duxbury advanced series in statistics and decision sciences. Thomson Learning.

COLLINS, R. 2003. Mean-shift blob tracking through scale space. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003.*, vol. 2, II–234–40 vol.2.

CUI, J., ZHA, H., ZHAO, H., AND SHIBASAKI, R. 2005. Tracking multiple people using laser and vision. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2116–2121.

CURTIS, S., GUY, S., ZAFAR, B., AND MANOCHA, D. 2011. Virtual tawaf: A case study in simulating the behavior of dense, heterogeneous crowds. In *Proc. of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 128–135.

CURTIS, S., SNAPE, J., AND MANOCHA, D. 2012. Way portals: efficient multi-agent navigation with line-segment goals. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, 15–22.

DU TOIT, N., AND BURDICK, J. 2010. Robotic motion planning in dynamic, cluttered, uncertain environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 966–973.

EVENSEN, G. 2003. The ensemble kalman filter: theoretical formulation. *Ocean Dynamics 55*, 343–367.

FOD, A., HOWARD, A., AND MATARIC, M. 2002. A laser-based people tracker. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, 3024–3029.

FULGENZI, C., SPALANZANI, A., AND LAUGIER, C. 2007. Dynamic obstacle avoidance in uncertain environment combining pvos and occupancy grid. In *Proc. of the IEEE International Conference on Robotics and Automation*, 1610–1616.

GONG, H., SIM, J., LIKHACHEV, M., AND SHI, J. 2011. Multi-hypothesis motion planning for visual object tracking. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 619–626.

GUY, S. J., CHHUGANI, J., CURTIS, S., DUBEY, P., LIN, M., AND MANOCHA, D. 2010. PLEdestrians: a least-effort approach to crowd simulation. In *Symp. on Computer Animation*, 119–128.

GUY, S., KIM, S., LIN, M., AND MANOCHA, D. 2011. Simulating heterogeneous crowd behaviors using personality trait theory. In *Symp. on Computer Animation*, 43–52.

HARGREAVES, J., ANNAN, J., EDWARDS, N., AND MARSH, R. 2004. An efficient climate forecasting method using an intermediate complexity earth system model and the ensemble kalman filter. *Climate Dynamics 23*, 7, 745–760.

HELBING, D., AND MOLNAR, P. 1995. Social force model for pedestrian dynamics. *Physical review E 51*, 5, 4282.

HENRY, P., VOLLMER, C., FERRIS, B., AND FOX, D. 2010. Learning to navigate through crowded environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 981–986.

KARAMOUZAS, I., AND OVERMARS, M. 2010. A velocity-based approach for simulating human collision avoidance. In *Intelligent Virtual Agents*, Springer, 180–186.

KARAMOUZAS, I., HEIL, P., VAN BEEK, P., AND OVERMARS, M. 2009. A predictive collision avoidance model for pedestrian simulation. *Motion in Games*, 41–52.

KIM, S., GUY, S., LIU, W., LAU, R., LIN, M., AND MANOCHA, D. 2012. Predicting pedestrian trajectories using velocity-space

reasoning. In *The Tenth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*.

KÖSTER, G., TREML, F., AND GÖDEL, M. 2013. Avoiding numerical pitfalls in social force models. *Phys. Rev. E 87* (Jun), 063305.

KRATZ, L., AND NISHINO, K. 2011. Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 99, 1–1.

LERNER, A., FITUSI, E., CHRYSANTHOU, Y., AND COHEN-OR, D. 2009. Fitting behaviors to pedestrian simulations. In *Symp. on Computer Animation*, 199–208.

LIAO, L., FOX, D., HIGHTOWER, J., KAUTZ, H., AND SCHULZ, D. 2003. Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, IEEE, 723–728.

LIU, W., CHAN, A. B., LAU, R. W. H., AND MANOCHA, D. 2014. Leveraging long-term predictions and online-learning in agent-based multiple person tracking. *IEEE Transactions on Circuits and System for Video Technology*.

LUBER, M., STORK, J., TIPALDI, G., AND ARRAS, K. 2010. People tracking with human motion predictions from social forces. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 464–469.

MCLACHLAN, G. J., AND KRISHNAN, T. 2008. *The EM Algorithm and Extensions (Wiley Series in Probability and Statistics)*, 2 ed. Wiley-Interscience, Mar.

MEHRAN, R., OYAMA, A., AND SHAH, M. 2009. Abnormal crowd behavior detection using social force model. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition,CVPR*, 935–942.

PELECHANO, N., ALLBECK, J., AND BADLER, N. 2008. Virtual crowds: Methods, simulation, and control. *Synthesis Lectures on Computer Graphics and Animation 3*, 1, 1–176.

PELLEGRINI, S., ESS, A., SCHINDLER, K., AND VAN GOOL, L. 2009. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Proc. of the IEEE 12th International Conference on Computer Vision*, 261–268.

PETTRÉ, J., ONDŘEJ, J., OLIVIER, A. H., CRETUAL, A., AND DONIKIAN, S. 2009. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Symp. on Computer Animation*, 189–198.

PRADHAN, N., BURG, T., AND BIRCHFIELD, S. 2011. Robot crowd navigation using predictive position fields in the potential function framework. In *American Control Conference (ACC)*, 4628–4633.

REYNOLDS, C. W. 1999. Steering behaviors for autonomous characters. In *Game Developers Conference. http://www. red3d. com/cwr/steer/gdc99*.

RODRIGUEZ, M., ALI, S., AND KANADE, T. 2009. Tracking in unstructured crowded scenes. In *Proc. of the IEEE 12th International Conference on Computer Vision*, 1389–1396.

SCHADSCHNEIDER, A., KLINGSCH, W., KLÜPFEL, H., KRETZ, T., ROGSCH, C., AND SEYFRIED, A. 2009. Evacuation dynamics: Empirical results, modeling and applications. 3142–3176.

SCHULZ, D., BURGARD, W., FOX, D., AND CREMERS, A. 2003. People tracking with mobile robots using sample-based joint probabilistic data association filters. *The International Journal of Robotics Research 22*, 2, 99.

SNAPE, J., VAN DEN BERG, J., GUY, S., AND MANOCHA, D. 2011. The hybrid reciprocal velocity obstacle. *IEEE Transactions on Robotics (TRO) 27*, 4, 696–706.

SVENSTRUP, M., BAK, T., AND ANDERSEN, H. 2010. Trajectory planning for robots in dynamic human environments. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4293–4298.

TRAUTMAN, P., AND KRAUSE, A. 2010. Unfreezing the robot: Navigation in dense, interacting crowds. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 797–803.

TRAUTMAN, P., MA, J., MURRAY, R. M., AND KRAUSE, A. 2013. Robot navigation in dense human crowds: the case for cooperation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2153–2160.

TREUILLE, A., COOPER, S., AND POPOVIĆ, Z. 2006. Continuum crowds. In *ACM SIGGRAPH*, 1160–1168.

VAN DEN BERG, J., LIN, M., AND MANOCHA, D. 2008. Reciprocal velocity obstacles for real-time multi-agent navigation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 1928–1935.

VAN DEN BERG, J., PATIL, S., SEWALL, J., MANOCHA, D., AND LIN, M. 2008. Interactive navigation of individual agents in crowded environments. In *Symp. on Interactive 3D Graphics and Games (I3D)*.

VAN DEN BERG, J., GUY, S., LIN, M., AND MANOCHA, D. 2011. Reciprocal n-body collision avoidance. *Robotics Research*, 3–19.

WAN, E., AND VAN DER MERWE, R. 2000. The unscented kalman filter for nonlinear estimation. In *Proc. of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium. AS-SPCC.*, 153–158.

WILKIE, D., VAN DEN BERG, J., AND MANOCHA, D. 2009. Generalized velocity obstacles. In *Proceedings of the IEEE/RSJ international conference on Intelligent robots and systems*, IEEE Press, Piscataway, NJ, USA, IROS'09, 5573–5578.

WORK, D., TOSSAVAINEN, O., BLANDIN, S., BAYEN, A., IWUCHUKWU, T., AND TRACTON, K. 2008. An ensemble kalman filtering approach to highway traffic estimation using gps enabled mobile devices. In *47th IEEE Conference on Decision and Control, CDC 2008.*, IEEE, 5062–5068.

YAMAGUCHI, K., BERG, A., ORTIZ, L., AND BERG, T. 2011. Who are you with and where are you going? In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1345 –1352.

ZIEBART, B. D., RATLIFF, N., GALLAGHER, G., MERTZ, C., PETERSON, K., BAGNELL, J. A., HEBERT, M., DEY, A. K., AND SRINIVASA, S. 2009. Planning-based prediction for pedestrians. In *Proc. of the IEEE/RSJ international conference on Intelligent robots and systems (IROS)*, IEEE Press, Piscataway, NJ, USA, IROS'09, 3931–3936.